Interim Guidance for Defense Information Infrastructure (DII) Common Operating Environment (COE) Realtime Extension

Version 1.0

December 2000

Submitted by:

K Wheeler

KENNETH WHEELER
DII COE Chief Engineer

Approved by:

Dawn C Meyerriecks

DAWN C. MEYERRIECKS
Chief Engineering Executive (Information Processing)
Joint Information Engineering Organization
Defense Information Systems Agency

Interim Guidance
for
Defense Information Infrastructure (DII)
Common Operating Environment (COE)
Realtime Extensions

Version 1.0

December 2000

Prepared for:

Joint Information Engineering Organization
Defense Information Systems Agency

General: DISA is currently developing realtime extensions to the DII COE in order to support realtime command and control systems. The first release of DII COE with realtime extensions (DII COE RTE) will include a Realtime Toolkit (RTTK), which is a set of development and integration tools that runs initially on Solaris, and the Realtime Kernel Services (RTKS), a set of runtime services which run on the target platforms identified below. In addition, some infrastructure services and common support applications will operate in the DII COE RTE environment. Although the schedules are not linked, the DII COE RTKS and RTTK are planned for deployment in roughly the same timeframe as DII COE version 5.0. Once product stability has been demonstrated, DISA plans to merge the DII COE RTKS and DII COE version 5.0 into a single software baseline.

Target Platforms: The initial reference platforms for the RTKS are referenced in the "Installation Procedures and Software Version Description" document for the RTKS [reference 6]. DISA will validate the RTKS on <u>specific configurations</u> of these initial target runtime platforms.

Scope: This interim guidance applies to all software targeted to the DII COE RTE in its beta and initial production releases. The DII COE RTE extends the DII COE capability in three dimensions: extended support for predictable realtime performance, configuration/tailoring of the DII COE kernel, and system composition principally from static object libraries assembled in an integration environment that will often involve cross-development for realtime platforms. Although all three of these dimensions extend orthodox DII COE concepts, they are all expected to be included in the mainstream DII COE starting with version 5.0.

This guidance is effective immediately and supplements the following documents:

- Integration and Runtime Specification (I&RTS), Version 4.1, August 2000

Definitions: The introduction of realtime extensions to the DII COE requires that the terminology be precise in distinguishing between a development/integration environment and a runtime environment. While the DII COE is primarily concerned with ensuring a consistent runtime environment, it must in certain cases also consider the development and integration environments. One reason for this is that software destined for a runtime environment may need to be further processed in a development or integration environment before it is suitable for execution in the target system. (For example, a developer may create a set of reusable object libraries. Object libraries are not directly executable but must be linked together with a "main program" to create an executable that then becomes part of the runtime environment.) This is particularly true for realtime systems where considerable control is required at integration time to properly configure software and data destined for a realtime environment. Moreover, realtime systems do not typically allow new applications to be loaded "dynamically" at an operational site, but instead require that new applications go through a formal integration phase "at the factory" to ensure that the new applications do not adversely impact the realtime performance of the system.

<u>Segment</u>**:** *A collection of one or more software and/or data units most conveniently managed as a unit of functionality*. Segments are defined from the perspective of an operator, not a developer, and are generally defined to keep related units together so that functionality may be easily included or excluded. They are usually defined as functional pieces (e.g., a word processor) that make sense from a system administrator or system integrator perspective because segments are the lowest level components above the configurable kernel that can be included in, or excluded from, a target system. Segments contain "self-describing" information that is used by automated tools to detect and avoid potential conflicts with other segments. In the present DII COE, this information is contained in the segment's `SegDescrip` directory.

Extended Toolkit**:** *A segment that contains documentation, linkable or shared libraries, data, and other items required for use in an integration, development, and/or runtime environment.* Extended toolkits may also contain executables that will be used in a runtime environment. The term "extended toolkit" supercedes the term "software development kit" used in prior DII COE releases.

Runtime Segmen**t:** *A segment that has been stripped of extraneous files and directories that are not required for a runtime target system.* A runtime segment is a subset of an extended toolkit. It uses the same segment descriptors as the extended toolkit.

The relationship of these three terms is illustrated in Figure 1 below. When used without qualification, a segment refers to both extended toolkits and runtime segments. A segment in the current DII COE, whether referring to an extended toolkit or a runtime segment, contains a `SegDescrip` directory that describes the segment, utilizes the directory structure described in the *I&RTS*, and conforms to the other standards and conventions stipulated in the *I&RTS*. An extended toolkit refers to a segment that is being used at development or integration time. A runtime segment is an extended toolkit that has been stripped of extraneous items (documentation, object code, C header files, etc.) that are not required for the runtime environment, and if necessary, further processed to be suitable for the runtime environment.

Runtime segments are typically created by deleting unnecessary files and directories at integration time and then running the `MakeInstall` tool so that the segment can be loaded by COE installation tools. This is *not* intended to state a requirement that realtime systems must load their system via the COE runtime installation tools. COE runtime installation tools are required to be used only for those systems that allow "dynamic" application loading at operational sites. Also note that the definition of a segment is not dependent upon how it is packaged (e.g., `tar` format, `MakeInstall` format) but is instead defined by the functionality it contains.

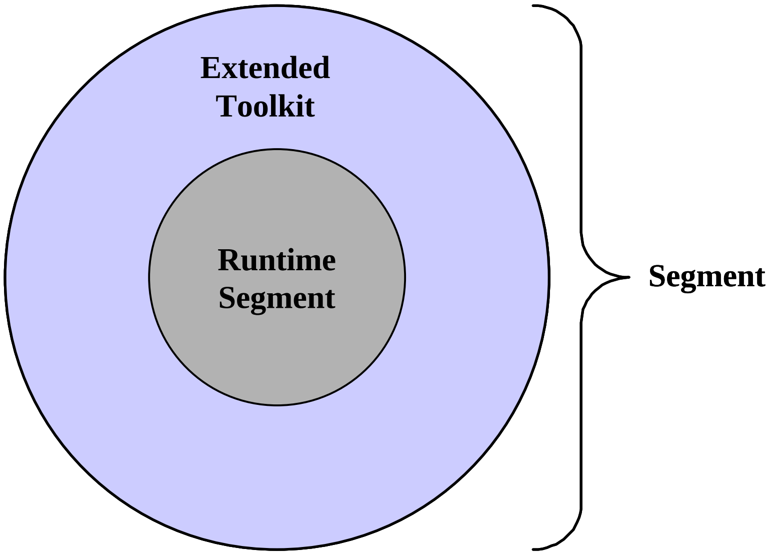

Figure 1: Relationship of Segment, Extended Toolkit, and Runtime Segment

Chief Engineer: A Chief Engineer is a senior technical position which embodies responsibility and authority for resources and the technical direction of a program or project. In this document, we will use the term "DII COE Chief Engineer" to identify the DISA DII COE Chief Engineer who leads the DII COE Engineering Office and holds this responsibility and authority for the DII COE. We use the term "Chief Engineer" (without qualification) to refer to a cognizant chief engineer position that is distinct from the DISA DII COE Engineering Office. Typically, this will be the Chief Engineer for a particular project or program. However, services and agencies may, at their discretion, assign this position to other offices as well. (Sometimes the Chief Engineer and the DII COE Chief Engineer may be the same person, depending on the role or activity being performed.)

Guidance:

1. When used in conjunction with the DISA DII COE RTKS, the target platforms referenced above will be DII COE RTE initial reference implementations.

2. Developers shall deliver software as either runtime segments or extended toolkits or in both formats. Developers are encouraged to design software so that either (a) the software is delivered in both formats, or else (b) the software is delivered as an extended toolkit but can be easily and mechanically converted to a runtime segment. All software must successfully pass the checks performed by VerifySeg. Software intended to be part of the DII COE (i.e., common support applications and infrastructure services) will be delivered as runtime segments unless otherwise approved by the DII COE Chief Engineer.

3. For software intended for a realtime platform, any assumptions, requirements, or constraints regarding scheduling policy, priority level of schedulable entities (threads, processes, etc.), use of shared resources, periods and other scheduling parameters should be identified, documented, and approved by the cognizant chief engineer. As a minimum, the scheduling policy and initial priority settings in the as-delivered software must be identified, documented and approved. This information shall be documented in the IntgNotes file.

4. The SegName descriptor file is required for all segments. For realtime-only segments, the "realtime" attribute is also required in this descriptor.

5. Requirements for shared memory to be reserved by the segment shall be specified in the `Hardware{` XE "`Hardware`" `}` descriptor using keyword $OS_SHARED_MEMORY.[1]

6. Restrictions on the platform, such as specific board options, shall be documented in the `Hardware{` XE "`Hardware`" `}` descriptor using keyword $CPU using an attribute of ":RESTRICTED." If the attribute is ":RESTRICTED," then the restrictions shall be documented in the IntgNotes file. (":RESTRICTED" is a new optional attribute for the $CPU keyword.)[2]

7. Dependencies on specific hardware devices shall be documented in the `Hardware{` XE "`Hardware`" `}` descriptor using keyword $HW_DEVICE.[3]

8. Restrictions on the operating system shall be documented in the `Hardware{` XE "`Hardware`" `}` descriptor using keyword $OPSYS using an attribute of

[1] The $OS_SHARED_MEMORY keyword is not yet supported.
[2] The ":RESTRICTED" attribute for the $CPU keyword is not yet supported.
[3] The $HW_DEVICE keyword is not yet supported.

":RESTRICTED." If the attribute is ":RESTRICTED," then the restrictions shall be documented in the IntgNotes file. (":RESTRICTED" is a new optional attribute for the $OPSYS keyword.) Requirements for documenting dependencies on other operating systems will be addressed on a case-by-case basis.[4]

9. For realtime-only segments that are designed to operate in less than a POSIX PSE54 environment, dependencies on operating system services should be documented in the RequiredServices descriptor using the keyword $POSIX_UNIT. Legal values for this keyword are identified in Appendix A of this Interim Guidance. (Legal values are based on POSIX Units of Functionality as described in POSIX 1003.13:1998.)[5]

10. For realtime-only segments, resource requirements of the software shall be documented in the IntgNotes file. These resource requirements include identification of any resources shared with other software or segments (including but not limited to critical sections), maximum holding times for all shared resources, and maximum blocking time attributable to the software. These resource requirements may include values of formulas governing CPU consumption, and the use of disk and network resources. Formulas should be provided to assist the integrator in tailoring gross resource requirements of the software to the specific mission environment. The methodology for determining these values shall be documented in the IntgNotes file.

11. All dependencies of the software (or its performance) on specific development tools or environments (e.g., compiler dependencies) shall be documented in the IntgNotes file.

12. For realtime-only segments, the POSIX 1003.13 profiles on which the software will run should be documented in the IntgNotes file.

13. For realtime-only segments, the developer should document in the IntgNotes file any known restrictions on the target environment that are essential to the effective execution of this software.

14. Segments shall fully specify direct dependencies upon kernel services. This is done through the `REQUIRES{ XE "Requires" }` segment descriptor exactly as would be done with dependencies on any other segment. [6]

References:
(1) Integration and Runtime Specification (I&RTS), Version 4.1, August 2000
(2) Real-time Segmentation Tutorial, Release 1.2, 7 June 2000
(3) Real-time Kernel Services SRS, Revision 1.3, 24 May 2000
(4) Real-time Extensions to DII COE Segment Development Tools SRS, Revision 1.0, 30 April 1999
(5) Real-time Extensions to Integration Tools of the DII COE, Revision 1.1, 8 May 2000
(6) Installation Procedures and Software Version Description (IP/SVD) for DII COE Real-time Toolkit, Version 0.0.1.1, Document Version 0.0.1.1, 5 May 2000
(7) 1996 (ISO/IEC) [IEEE/ANSI Std 1003.1, 1996 Edition] Information Technology—Portable Operating System Interface (POSIX&)—Part 1: System Application Program Interface (API) [C Language] *This edition incorporates extensions for realtime applications (1003.1b-1993, 1003.1I-1995) and threads (1003.1c-1995)*
(8) 1003.13-1998 IEEE Standard for Information Technology—Standardized Application Environment Profile (AEP)—POSIX Realtime Application Support

[4] The ":RESTRICTED" attribute for the $OPSYS keyword is not yet supported.
[5] The $POSIX_UNIT keyword is not yet supported.
[6] The implementation of this capability has not yet been defined.

Appendix A: Valid Keyword Definitions for POSIX Units of Functionality

This appendix establishes the legal values for the $POSIX_UNIT keyword. These values are based on POSIX Units of Functionality and are defined in Table A-1 and A-2.

Table A-1 lists POSIX units of functionality as laid out in POSIX.13. A segment which is targeted for an environment which is less than a fully functional PSE 54 environment (in other words, less than "full POSIX") would document their dependencies on operating system functionality in terms of these POSIX Units of Functionality using the values in the table.

The unit POSIX_C_LANG_SUPPORT is a special case. POSIX_C_LANG_SUPPORT is informally subdivided into six subcategories by POSIX.13. Table A-2 provides a subdivision of the POSIX.13 UoF, POSIX_C_LANG_SUPPORT, which can optionally be used by DII COE RTE segments that wish to provide more specific dependencies within the POSIX.13 UoF POSIX_C_LANG_SUPPORT. For example, the POSIX_C_LANG_MATHEMATICS operations are called out as a separate option, to permit segments to establish dependencies on high-performance mathematics libraries that may not otherwise be provided by the DII COE RTE kernel/operating system implementation. Use of setjmp()/longjmp() often has substantial impact on the performance and analyzability of C software, thus it is helpful to indicate that a particular segment depends on this part of the C language, via POSIX_C_LANG_NON-LOCAL_JUMPS. The operations in POSIX_C_LANG_GENERAL_UTILITIES are normally provided by the C language library, with relatively minor impact on software performance, portability and analyzability.

To reiterate: The developer of a segment which depends on POSIX_C_LANG_SUPPORT may elect to either:

1. Identify dependencies on the detailed subcategories listed in Table A-2, or
2. Identify a dependency only on POSIX_C_LANG_SUPPORT, which implies dependency on POSIX_C_LANG_GENERAL_UTILITIES, POSIX_C_LANG_MATHEMATICS and POSIX_C_LANG_NON-LOCAL_JUMPS.

Table A-1. Legal Values for the $POSIX_UNIT Keyword

POSIX.1 & POSIX.5b Unit of Functionality Requirements/Options (IEEE P1003.13-1998 March 1998)

POSIX_SINGLE_PROCESS
POSIX_MULTI_PROCESS
POSIX_JOB_CONTROL
POSIX_SIGNALS
POSIX_USER_GROUPS
POSIX_FILE_SYSTEM
POSIX_FILE_ATTRIBUTES
POSIX_FIFO
POSIX_DEVICE_IO
POSIX_FD_MGMT
POSIX_PIPE
POSIX_DEVICE_SPECIFIC
POSIX_SYSTEM_DATABASE
POSIX_C_LANG_SUPPORT
POSIX_ADA_LANG_SUPPORT

POSIX.1 Option Requirements

_POSIX_JOB_CONTROL
_POSIX_SAVED_IDS
_POSIX_CHOWN_RESTRICTED
_POSIX_NO_TRUNC
_POSIX_VDISABLE
NGROUPS_MAX

POSIX.5b Option Requirements

Job_Control_Support
Saved_IDs_Support
Change_Owners_Restriction
Filename_Truncation

POSIX.1b Option Requirements

_POSIX_ASYNCHRONOUS_IO
_POSIX_MAPPED_FILES
_POSIX_MEMLOCK
_POSIX_MEMLOCK_RANGE
_POSIX_MEMORY_PROTECTION
_POSIX_MESSAGE_PASSING
_POSIX_PRIORITIZED_IO
_POSIX_PRIORITY_SCHEDULING
_POSIX_REALTIME_SIGNALS
_POSIX_SEMAPHORES
_POSIX_SHARED_MEMORY_OBJECTS
_POSIX_SYNCHRONIZED_IO
_POSIX_TIMERS
_POSIX_FSYNC

POSIX.1c Option Requirements

_POSIX_THREADS
_POSIX_THREAD_ATTR_STACKSIZE
_POSIX_THREAD_PRIO_INHERIT
_POSIX_THREAD_PRIORITY_SCHEDULING
_POSIX_THREAD_PRIO_PROTECT
_POSIX_THREAD_PROCESS_SHARED
POSIX THREAD_ATTR_STACKADDR
_POSIX_THREAD_SAFE_FUNCTIONS

POSIX.1c - Unit of Functionality Req's

POSIX_USER_GROUPS_R
POSIX_DEVICE_SPECIFIC_R
POSIX_FILE_LOCKING
POSIX_C_LANG_SUPPORT_R
POSIX_ SYSTEM_DATABASE_R

Table A-2. Subdivision of POSIX_C_LANG_SUPPORT

RT DII COE Unit of Functionality Name
POSIX_C_LANG_GENERAL_UTILITIES
POSIX_C_LANG_NON-LOCAL_JUMPS
POSIX_C_LANG_MATHEMATICS
POSIX_C_LANG_CHARACTER_HANDLING
POSIX_C_LANG_STRING_HANDLING
POSIX_C_LANG_DATE_AND_TIME

Appendix B: Compliance Checklists

Note: This *Interim Guidance Appendix B* is issued as an aid to the realtime community for developers, system engineers, and program managers who are currently developing, or are planning to develop, DII COE-based applications targeted for realtime environments. This *Interim Guidance* presents a compliance checklist that incorporates necessary considerations for realtime applications. The definition of compliance levels, however, remains unchanged.

The terms segment, extended toolkit, and runtime segment are used in the discussion that follows. These terms are formally defined elsewhere in the *I&RTS*. Briefly, an extended toolkit is a segment that is used in a development or integration environment. The term "extended toolkit" supersedes the "software development kit" (SDK) terminology used in previous DII COE releases. A runtime segment is a segment that is used in a target system and that has been stripped of files and directories that are used only in the development or integration environment. Unless otherwise qualified, the term segment refers to both extended toolkits and runtime segments.

DII COE compliance is the cornerstone to ensure seamless ~~segment~~ integration and proper system operation. An objective technique for measuring DII COE compliance is required. Such a system has the following advantages:

- It allows quantitative statements to be made about whether or not a segment or COE-based system is compliant. It also provides an objective measure of the *degree* to which a segment or system is DII COE-compliant.

- It serves to identify areas in which ~~segments need to~~ improvement is necessary to achieve compliance. A side benefit is the identification of potential areas of interoperability problems because of identifying areas where the system overlaps COE functionality.

- It provides a meaningful way to quantitatively compare segments in addition to traditional measures such as functional coverage. One important dimension to this technique is that it directly incorporates interoperability comparisons between segments being evaluated.

- It aids in developing a migration strategy for legacy systems. Program managers can use the checklist to determine where they stand with regards to compliance and then use the identified areas of non-compliance to create a strategy and schedule for achieving the target level of compliance.

This appendix addresses primarily Category 1 (Runtime Environment) compliance, although there are also certain Category 2 (Style Guide) compliance items[1] listed. Chapter 2 defines Category 1 compliance as eight levels of progressively deeper integration, because compliance cannot be an all-or-nothing proposition for legacy systems. The levels progress from a state of "peaceful coexistence" to "federation of systems" to true integration. Chapter 2 also shows how the levels of compliance map to levels of interoperability, and that interoperability increases as the level of DII COE compliance increases. As noted in Chapter 2, segments shall achieve Level 7 compliance before being accepted or advertised as an approved DISA product. At DISA's discretion, segments that are Level 5 or Level 6 compliant may be accepted as prototypes and fielded at selected sites for evaluation purposes. Developers must achieve Level 4 compliance before DISA will consider evaluating a prototype for eventual migration into the COE or COE-based DISA systems.

This appendix contains a series of questions, in a checklist format, which are organized by compliance level. The philosophy behind this design for the compliance checklists is to begin with an agreement on a set of standards[2], ensure non-interference when installed on the same LAN, then non-interference when installed on the same platform, and finally to interoperability through sharing the same software and data. With this strategy, it is possible to define a minimally acceptable level of compliance that balances system risk against cost to achieve full compliance.

Segments shall be evaluated against these checklists to determine the degree of compliance. There are several things to note to properly apply the compliance checklists in this appendix.

1. System components, applications, and data are not required to be in segment format until Level 4. Therefore, the compliance checklists use the generic term "application" to refer to any software or data component being evaluated in levels 1-3. Beginning with Level 4, they are properly referred to as segments because compliance at Level 4 requires full or abbreviated segmentation.

2. Each item in the checklists shall be answered as *True*, *False*, or *Not Applicable* as appropriate. Not all questions are necessarily applicable to all segments, and hence should be marked as "N/A." For example, questions that deal with database issues are not applicable to applications/segments that do not use a database.

3. The Category 1 compliance level assigned to the application/segment is the highest numbered level for which there are no "False" replies. As explained in Chapter 2, it is not permissible to describe an application/segment in a manner such as "80% Level 6" compliant.

[1] The *DII COE User Interface Specifications***{** XE "*User Interface Specification*" **}** addresses Category 2 compliance requirements and how they map to the equivalent Category 1 levels presented here.

[2] The JTA**{** XE "JTA" **}** contains a standards profile that compliant systems must adhere to. However, to allow for migration of legacy systems, full compliance is not stipulated in Level 1 but is spread judiciously across all 8 levels.

4. Note that each question is prefaced by the level to which it applies. Thus, question 6-3 refers to the third compliance question for Level 6.

5. Each question in the checklists is independent of the operating environment (e.g., UNIX versus NT) unless otherwise noted. Environment-specific questions are noted by including “(UNIX),” ~~or ~~“(NT),” or “(Realtime)” as appropriate at the beginning of the question, and occasionally by denoting specific parts of the question that are applicable only to UNIX,~~ or~~ NT, or Realtime~~ specific~~.

6. Each successive level is inclusive. For example, Level 6 compliance means that the segment has achieved compliance for levels 1 through 5.

7. In the checklists there are references made to approvals by the Chief Engineer. Unless otherwise qualified, this means the DII COE Chief Engineer for COE-component segments and the cognizant DOD program Chief Engineer for mission-application segments. The principle being applied is that any modification that affects interoperability or the COE requires approval by the DII COE Chief Engineer. All other approvals are the domain of the program Chief Engineer because they are limited in scope to the system being built.

8. In the checklists there are references to the SSA. Unless otherwise qualified, this means the DII COE SSA for COE-component segments and the cognizant DOD program SSA for mission-application segments.

9. In each case, it is the intent that “sanity is to prevail.” Applications/segments are expected to comply with the requirements specified, but the DII COE Chief Engineer may grant unusual exceptions on a case-by-case basis.

10. Validation requirements that use the phrase “execute correctly” (or similar phrases) should be interpreted as meaning “functionally correct” without implying correctness with respect to performance.

11. The term “component,” unless qualified as COE-component segment, refers to both COE software and mission applications executing on a DII COE platform.

The checklists presented in this appendix are organized in a very deliberate way and with a deliberate objective in mind of evaluating DII COE compliance. There are several points about how the checklists are organized.

- The checklists are organized in such a way as to evaluate individual segments. However, as Chapter 2 describes, the compliance of individual segments can be combined into a composite compliance level. Many of the items in the appendix are applicable to only specific types of segments (e.g., COE-component segments, database segments), and should be noted as “not applicable” for many segments.

- The organization presented here was chosen so that the compliance criteria for a segment is independent of how the segment will ultimately be used in the target system.

- The organization presented here does not directly address the amount of work required to reach any specific level of compliance. The reason is because the effort required to reach compliance is heavily dependent upon the system under evaluation, and the areas in which the system is weak with respect to compliance.

- The organization presented here is conducive to pinpointing problem areas both from a compliance perspective, and as a side effect, from an interoperability perspective. Part of the effort in achieving compliance is to ensure conformance to standards, and to ensure that there is no duplication of COE services. This directly leads to pinpointing potential interoperability problem areas.

- The COE specifically avoids weighting compliance questions. This has the obvious disadvantage of assigning equal importance to each question within a level, but the reverse problem is that assigning weights cannot be done outside the context of the objectives of the end system.

- The checklist is *not* intended to be organized as a requirements traceability matrix. Therefore, there is no attempt to ensure that each checklist item is traceable back to one or more specific sections in the main body of this document. However, it is intended that each requirement listed in the main body of the document be represented by one or more checklist items. Organizing the checklist as a requirements traceability matrix is under consideration for a future release.

- The DII COE requires adherence to Microsoft Logo requirements for ~~well behaved~~well-behaved Windows NT-based applications. Microsoft Logo requirements are not listed separately here except: (a) when requirements are considered to be so critical to the success of the DII COE that they warrant special mention, or (b) when DII COE requirements are different than Microsoft Logo requirements.

~~This release of the *I&RTS* introduces several new requirements in areas, especially in security, which were not adequately addressed in previous releases. The importance of this is that segments that were compliant at a particular level under a previous *I&RTS* release are not necessarily still compliant at the same level under this *I&RTS* release. This change has been necessitated mostly by a need to improve the security posture of COE-based systems.~~

~~In general, applications should still be able to reach Level 5 compliance without requiring source code changes. But this is true only if good programming practices and good security practices have been followed. For example, applications with "world" permissions will likely require modification to meet Level 5 compliance criteria (see the *Security* series of questions below under Level 5 compliance).~~

~~Due to editorial changes made in this release of the *I&RTS*, compliance items have been renumbered from the previous release. New compliance requirements have been added in the following situations:~~

1. ~~Some compliance questions have been split into 2 requirements for the sake of clarity.~~

2. ~~Some compliance items have been added to the checklists to cover requirements listed in the main body of the document, especially in the database section, but overlooked in forming the checklists for the previous release.~~

3. ~~Some checklist items have been moved from a higher compliance level to a lower compliance level to improve the security posture of COE-based systems.~~

4. ~~Checklist items have the same number (e.g., 6-3, 7-10) in this release of the *I&RTS* as they had in the previous release. Checklist entries that have been deleted are so noted without actually deleting the item and then renumbering all subsequent items. For example, see item 5-15. New entries are added where they are most appropriate and are numbered by adding a letter to the number for the item preceding the new checklist entry. For example, see 5-16a and 5-16b, which have been inserted into the checklist immediately following checklist item 5-16.~~

B-1. Standards Compliance (Level 1)

Operating System Services	
T F N/A 1-1	The operating system ~~is configured to~~ supports TCP/IP protocols. Systems that do not require TCP/IP may configure it out if the operating system is so configurable.
T F N/A 1-2	The operating system ~~is configured to~~ supports UDP protocols. Systems that do not require UDP may configure it out if the operating system is so configurable.
T F N/A 1-3	The operating system ~~is configured to~~ supports SLIP and/or PPP. Systems that do not require SLIP and/or PPP may configure them out if the operating system is so configurable.
T F N/A 1-4	Custom device drivers added to support program-unique requirements, if any, do not interfere with native capabilities of the operating system nor do they cause a violation of other mandated standards for the operating system or network.
T F N/A 1-5	(UNIX) The operating system and associated software conform to the following standards from the *JTA***{** XE "JTA" **}**: (a) *ISO 9945-1:1996, Information Technology - Portable Operating System Interface for Computer Environment (POSIX) - Part 1: System Application Program Interface (API) [C Language]*, as profiled by *FIPS 151-2:1994*. (b) *IEEE 1003.1g, D6.6: March 1998, Standard for Information Technology - Portable Operating System Interface (POSIX) - Part 1: System Application Program Interface - Protocol Independent Interfaces (PII) [C Language]*.
T F N/A 1-6	(NT) The operating system and associated software conform to the following standard from the *JTA*: *Win32 APIs*, *Window Management and Graphics Device Interface, Volume 1 Microsoft Win32 Programmers Reference Manual*, 1993 or later, Microsoft Press.
T F N/A 1-7	(NT) The operating system is one of the following commercially licensed versions of Windows NT, as used by the COE, or higher, or a licensed version of NT that has been validated through the Kernel Platform Compliance program: (a) Windows NT Workstation, (b) Windows NT Server, or (c) Windows NT Server, Enterprise Edition.
T F N/A ~~1-7~~1-7a	(Realtime) The RTOS shall provide, as a minimum, all of the capabilities associated with profile PSE54 (Multi-Purpose Realtime System Profile) as defined in *IEEE 1003.13, POSIX Realtime Application Support (AEP).*

Network Services				
T	**F**	**N/A**	**1-8**	The application, if ~~required~~it uses TCP/IP, uses only those TCP/IP interfaces ~~provided~~ packaged by the ~~native~~ operating system vendor.
T	**F**	**N/A**	**1-9**	The application, if ~~required~~it uses UDP or point-to-point interfaces, uses only those UDP or point-to-point interfaces ~~provided~~ packaged by the ~~native~~ operating system vendor.
T	**F**	**N/A**	**1-10**	The application, if ~~required~~it uses SLIP or PPP interfaces, uses only those SLIP or PPP interfaces ~~provided~~ packaged by the ~~native~~ operating system vendor.
GUI Environment				
T	**F**	**N/A**	**1-11**	The application complies with the style of the native GUI (e.g., Motif for UNIX, Windows for NT). (See GUI compliance requirements in the *DII COE User Interface Specifications*{ XE "*User Interface Specification*" }.)
T	**F**	**N/A**	**1-12**	(UNIX) The windowing environment conforms to the following standard from the *JTA*{ XE "*JTA*" }: *ISO 9945-2: 1993, Information Technology* - *Portable Operating System Interface for Computer Environments (POSIX)* - *Part 2: Shell and Utilities* as profiled by *FIPS PUB 189:1994*.
T	**F**	**N/A**	**1-13**	(UNIX) The windowing environment conforms to the X Windows and Motif standards~~ specified in the *JTA*~~{ XE "*JTA*" }.
Database Services				
T	**F**	**N/A**	**1-14**	If an RDBMS is used, it supports *FIPS-127-2* SQL queries.

B-2. Network Compliance (Level 2)

Security Services	
T F N/A 2-1	(UNIX) The application is able to operate correctly with the operating system security modules enabled (BSM for Solaris, C2 enabled for HP, etc.) and under the constraints imposed by the UNIX kernel security configuration (“lockdown”) settings. Refer to the latest version of the *DII COE Security Features Developers Guide* for specifics about these settings. Exceptions have been brought to the Chief Engineer for resolution. Waivers, if any, have been documented accordingly.
T F N/A 2-2	(NT) The application can perform under the constraints imposed by the NT kernel security configuration (“lockdown”) settings and the security templates installed as part of the NT baseline security configuration. Refer to the latest version of the *DII COE Security Features Developers Guide* for specifics about these settings. Exceptions have been brought to the Chief Engineer for resolution. Waivers, if any, have been documented accordingly.
Operating System Services	
T F N/A 2-3	(UNIX) The operating system supports NFS servers and clients.
T F N/A 2-4	(UNIX) The operating system can be configured to support DNS/NIS+. (Note: The requirement is that the operating system be capable of supporting centralized management of key resources such as hostnames, user accounts, etc. NIS+ is not a specific requirement because not all vendors support it.)
T F N/A 2-5	(NT) Windows NT is configured to use the NTFS file system for files stored on hard disks. (Note: NT uses the FAT file system for floppy diskettes and CDFS for CD-ROMs. Such usage is generally transparent to applications. However, NTFS is required on the hard disks for security reasons.)
Network Services	
T F N/A 2-6	The operating system supports sockets. For Unix, this includes supporting Berkeley sockets. For NT, this includes supporting Winsock 2.0.
T F N/A 2-7	The application is able to operate properly in an environment where other applications are performing UDP broadcasts.
T F N/A 2-8	The application does not require any particular hostname conventions nor does it need reserved IP addresses.
T F N/A 2-9	The ability of the application to execute correctly is independent of the type of network address space implemented for the LAN (e.g., Class B or Class C) connected to the platform.

T	**F**	**N/A**	**2-10**	(UNIX) The application can operate in a DNS/NIS+ environment. (Note: The requirement is that the application be able to operate correctly when the features supported by the operating system for centralized management of key resources are enabled.)
T	**F**	**N/A**	**2-11**	(NT) The application can operate correctly in an environment that includes Microsoft domains and workgroups.
GUI Environment				
T	**F**	**N/A**	**2-12**	(UNIX) If the application is an X Windows application, it is compatible with the X server supplied by the COE.
Database Services				
T	**F**	**N/A**	**2-13**	Database updates operate correctly with DBMS security audits enabled.
T	**F**	**N/A**	**2-14**	The database is recoverable and gracefully handles DBMS server, network, and client application failures. This includes both hardware and software failures.
T	**F**	**N/A**	**2-15**	RDBMS products used implement strict two-phase locking**{** XE "two-phase locking" **}** for database transactions.

B-3. Platform Compliance (Level 3)

Operating System Services				
T	**F**	**N/A**	**3-1**	If extensions to the operating system as configured for the COE are required, all such extensions have been identified, documented, and approved by the ~~DII COE~~cognizant Chief Engineer. This includes the configuration of all operating system resources including: (a) for UNIX, the amount of shared memory required, the number of semaphores, the message queue size, etc. (b) for NT, registry entries used to configure the operating system and core components.
T	**F**	**N/A**	**3-2**	The operating system configuration required by the application does not decrease or conflict with any system resources (shared memory allocated, number of semaphores allowed, message queue size, etc.) as already configured for the COE. The application may increase system resource configurations, but not decrease them.
T	**F**	**N/A**	**3-3**	The application does not use hardcoded port assignments and is not sensitive to specific ports other than well-known port assignments (e.g., `ftp`, `ping`, and ports less than 1024). If the application uses network services, including standard services such as `ftp` and `ping` as well as its own private services, it retrieves the standard service port number(s) by service name.
T	**F**	**N/A**	**~~3-3~~3-3a**	(Realtime) For components intended for a realtime platform, any assumptions, requirements, or constraints regarding scheduling policy, priority level of schedulable entities (threads, processes, etc.), and use of shared resources should be identified, documented, and approved by the cognizant Chief Engineer. As a minimum, the scheduling policy and initial priority settings in the as-delivered software shall be identified, documented, and approved.
Network Services				
T	**F**	**N/A**	**3-4**	If the application uses `ftp`, it can operate in an environment where only anonymous `ftp` is available.
GUI Environment				
T	**F**	**N/A**	**3-5**	(UNIX) The application does not make direct calls to X libraries that conflict with applications that use Motif libraries to access lower level X functions. For example, the application does not use lower-level X library functions to establish window border style or colors that either conflict with or override settings established by Motif.
T	**F**	**N/A**	**3-6**	(UNIX) The application does not alter any files in the vendor-supplied X or Motif directories (e.g., modify `rgb.txt` or `Xdefaults`) unless approved by the DII COE Chief Engineer. Approval by the DII COE Chief Engineer is required because of the potential effect of the application on other segments running on the same platform.
T	**F**	**N/A**	**3-7**	(UNIX) The application uses the same X server version that is supplied by the COE.

T	**F**	**N/A**	**3-8**	(UNIX) The application uses either the same version of Motif as provided by the COE or does a static link to Motif libraries so that it does not conflict with other COE-based segments.
T	**F**	**N/A**	**3-9**	(NT) Unless a COTS application, the application uses only Win32 APIs to access Windows routines.
Database Services				
T	**F**	**N/A**	**3-10**	The application does not modify the user's DBMS environment that is established by the DBMS COE-component segment.
COTS Products				
T	**F**	**N/A**	**3-11**	The software runs in a well-behaved manner in an environment that includes DII COE approved COTS products and segments, as specified in the *DII COE Buildlist Worksheet* (available on the DISA DII COE website, CM subpage) for the COE version being used.
T	**F**	**N/A**	**3-12**	Configuration changes made to COTS products, if any, do not render inoperable any features normally available with the COTS product. All configuration changes shall be backward compatible.
T	**F**	**N/A**	**3-13**	The application does not require any source code modifications to COTS products, except as approved by the DII COE Chief Engineer.
T	**F**	**N/A**	**3-14**	(NT) If the application is a COTS product that uses 16-bit APIs, there is no 32-bit alternative.
Runtime Environment				
T	**F**	**N/A**	**3-15**	The application does not alter any files outside its own directory in such a way that it conflicts with any other COE-based segment.
T	**F**	**N/A**	**3-16**	The application can operate on a COE-configured platform without altering the location or version of any system software.
Miscellaneous				
T	**F**	**N/A**	**3-17**	(NT) At a minimum, the application supports 16x16, 32x32, and 48x48 icons.

B-4. Bootstrap Compliance (Level 4)

Security Services				
T	**F**	**N/A**	**4-1**	If the application is a collection of subcomponents with different security levels, the security level of the application is given as the highest security level for any component within the application.
T	**F**	**N/A**	**4-2**	Documentation is submitted with the segment that clearly identifies releasability restrictions.
Standards Compliance				
T	**F**	**N/A**	**4-3**	All software and data are packaged as segments through either the full or abbreviated segmentation process.
T	**F**	**N/A**	**4-4**	Whether full or abbreviated segmentation is used, the segment uses ~~the~~ a configured COE kernel provided by the COE, or all extensions required are documented and handled by the segment in such a way that it does not interfere with other segments. For example, the segment does not destructively overwrite community files**{** XE "community files" **}** because other segments may also need to make alterations to the community file during their own installation.
T	**F**	**N/A**	**4-5**	For full segmentation ~~segments~~, the runtime segment can be installed and removed completely through the COE installation tools. (Note that modifications made by the installation tools for the `COEServices`**{** XE "`COEServices`" **}** descriptor will not necessarily be reversed during segment removal, as per design. This requirement does stipulate, however, that segment developers shall reverse modifications that were made directly through `Community`**{** XE "`Community`" **}**, `PostInstall`**{** XE "`PostInstall`" **}**, or `PreInstall`**{** XE "`PreInstall`" **}** descriptors.) If the runtime segment is a “permanent” segment (i.e., it has no `DEINSTALL`**{** XE "`DEINSTALL`" **}** file. See the Runtime Environment chapter) and is not a candidate for removal, the runtime segment has been tested to ensure that upgrades successfully preserve data files that must be retained during upgrades.
T	**F**	**N/A**	**4-6**	For abbreviated segmentation ~~segments~~, the segment descriptor files can be installed and completely removed through the COE installation tools while the runtime segment itself can be completely removed using the vendor-provided installation/deinstallation functions. If the runtime segment is a “permanent” segment**{** XE "`DEINSTALL`" **}** and is not a candidate for removal, the runtime segment has been tested to ensure that upgrades successfully preserve data files that must be retained during upgrades.
T	**F**	**N/A**	**4-7**	For abbreviated segmentation segments, the cognizant Chief Engineer has granted prior approval to do abbreviated segmentation and has approved the installation methodology and sequencing.
T	**F**	**N/A**	**4-8**	(NT) The segment does not modify the root-level `AUTOEXEC.BAT`**{** XE "`AUTOEXEC.BAT`" **}**, `CONFIG.SYS`**{** XE "`CONFIG.SYS`" **}**, `AUTOEXEC.NT`**{** XE "`AUTOEXEC.NT`" **}**, or `CONFIG.NT`**{** XE "`CONFIG.NT`" **}** files.
T	**F**	**N/A**	**4-9**	(NT) The segment does not modify `WIN.INI`**{** XE "`.INI`" **}{** XE "`WIN.INI`" **}** and `SYSTEM.INI`**{** XE "`SYSTEM.INI`" **}**. The segment may freely modify its own local `.INI` files, if it has any.

Database Services				
T	**F**	**N/A**	**4-10**	The segment does not modify the core, public, or system database storage areas (e.g., create rollbacks in Oracle System tablespace).
COTS Products				
T	**F**	**N/A**	**4-11**	The segment uses the same COTS version and configurations as those specified by the applicable *DII COE Buildlist Worksheet* (available on the DISA DII COE website, CM subpage) for any COTS product it uses that may also reside on the platform. This includes the requirement to use the JRE supplied by the COE unless an alternative is approved by the DII COE Chief Engineer.
Runtime Environment				
T	**F**	**N/A**	**4-12**	Runtime extensions to the COE required by the segment have been identified and documented.
T	**F**	**N/A**	**4-13**	The segment uses the same global runtime environment configuration as provided by the COE, or for realtime systems, as configured by a DII COE-approved process. Extensions to the global environment, if any, are made through applicable segment descriptors or Win32 APIs (for NT). Exceptions have been brought to the Chief Engineer for resolution. Waivers, if any, have been documented accordingly.
T	**F**	**N/A**	**4-14**	The segment uses the same versions, configurations, patches, and file locations as provided by the COE for all components of the COE kernel.
T	**F**	**N/A**	**4-15**	The segment uses the DII COE directory layout or a migration plan to achieve proper directory layout has been prepared.
T	**F**	**N/A**	**4-16**	(NT) The segment is able to handle UNC and long filenames/directory names.

B-5. Minimal DII COE Compliance (Level 5)

Security Services				
T	**F**	**N/A**	**5-1**	~~If~~ For an aggregate segment, the security level of the parent segment dominates the security level of the child segments.
T	**F**	**N/A**	**5-2**	For COE-component segments, if the segment provides a command-line mode or feature, the DII COE Chief Engineer has granted prior approval.
T	**F**	**N/A**	**5-3**	For mission-application segments, if the segment provides a command-line mode or feature, the Chief Engineer has granted prior approval.
T	**F**	**N/A**	**5-4**	The segment does not provide a "back door" access to a command-line prompt. If a command-line mode is available, it is through a known, documented approach for all authorized users and not through some hidden, undocumented approach.
T	**F**	**N/A**	**5-5**	For all segments, whether COE-component segments or mission-application segments, prior approval has been granted by the Chief Engineer to provide a command-line mode or feature that allows "superuser" access.
T	**F**	**N/A**	**5-6**	Entering a command-line mode requires the operator to enter a password and forces execution of the system login process.
T	**F**	**N/A**	**5-7**	(UNIX) If privileged user permissions are required during segment installation or removal**{** XE "`$ROOT`" **}**, the Chief Engineer has granted prior approval.
T	**F**	**N/A**	**5-8**	The segment contains no directories or files, nor does it create directories or files, at install time or runtime that grant world write permissions. `VerfiySeg` will generate warnings, which must be explained in `VSOutput`**{** XE "`VSOutput`" **}** and the *SVD* or equivalent document, for any segment that violates this requirement.
T	**F**	**N/A**	**5-9**	The segment does not insert the current working directory (e.g., "`.`") nor "~" into the runtime search path for executables. `VerifySeg` will generate warnings, which must be explained in `VSOutput`**{** XE "`VSOutput`" **}** and the *SVD* or equivalent document, for any segment that violates this requirement. This rule does not apply to `PostInstall`, `PreInstall`, or `DEINSTALL` descriptors or other scripts that are used only during the installation/deinstallation process, nor to scripts used only in a software development environment.
T	**F**	**N/A**	**5-10**	The segment contains only subdirectories directly underneath the segment's home directory. No files are contained in the segment's home directory.
T	**F**	**N/A**	**5-11**	(UNIX) The segment uses GIDs that were assigned at segment registration time. The GID is segment specific, based on the segment prefix, and is used ~~by the segment~~ to restrict unauthorized access to the segment or its data.
T	**F**	**N/A**	**5-12**	(UNIX) If the segment uses the COE GID, the DII COE Chief Engineer has approved such usage.

T	**F**	**N/A**	**5-13**	(UNIX) The segment does not alter the `umask{ XE "umask" }` setting[3] established by the COE for the runtime environment.
T	**F**	**N/A**	**5-14**	(UNIX) The segment does not contain or create any shell scripts that `SUID` or `SGID` to root. `VerifySeg` will generate warnings, which must be explained in `VSOutput{ XE "VSOutput" }` and the *SVD* or equivalent document, for any segment that violates this requirement.
T	**F**	**N/A**	**5-15**	**This requirement has been deleted.**
T	**F**	**N/A**	**5-16**	(NT) The segment has no directories or files that grant permissions to the “Everyone,” “Guest,” or “Domain Guests” groups.
T	**F**	**N/A**	**5-16a**	(NT) Files of the type `*.exe`, `*.com`, `*.cmd`, `*.bat`, `*.dll`, `*.hlp`, and `*.inf` within subdirectories of the segment’s home directory may not be modified by Authenticated Users.
T	**F**	**N/A**	**5-16b**	(NT) Within the `Program Files` directory structure, Authenticated Users will be granted no more than Read (RX) access to files of the type `*.exe`, `*.com`, `*.cmd`, `*.bat`, `*.dll`, `*.hlp`, and `*.inf`.
Standards Compliance				
T	**F**	**N/A**	**5-17**	The runtime segment uses the same COE kernel (either unchanged or configured by a DII COE-approved configuration process) as provided by the COE and documented in the applicable *DII COE Buildlist Worksheet* (available on the DISA DII COE website, CM subpage) for the COE version being used.
T	**F**	**N/A**	**5-18**	All directory and filenames contain only printable, non-blank, standard ASCII characters.
T	**F**	**N/A**	**5-19**	The segment does not create user login accounts. (This does not restrict segments from creating “non-login accounts” for use in establishing a ~~segment~~ group id.)
T	**F**	**N/A**	**5-20**	The runtime segment can operate in an environment where user accounts are created and deleted at any time by the site administrator responsible for managing user accounts. The runtime segment accounts for this and creates and initializes operator preferences the first time the segment is activated by the new user after ~~a~~ the new account is created.
T	**F**	**N/A**	**5-21**	The segment loads correctly into the directory selected by the person performing the installation. It does not require being loaded in any specific directory unless the Chief Engineer has granted a waiver. (This requirement does not apply to COTS segments.)
T	**F**	**N/A**	**5-22**	The segment conforms, in its `VERSION` descriptor, to the COE version numbering scheme.

[3] This does not apply to a temporary `umask` setting established during installation as long as the temporarily established `umask` setting is no longer in effect once the segment has been installed.

T	**F**	**N/A**	**5-23**	The runtime segment does not move directories or files from the segment's home directory into other directories unless approved by the DII COE Chief Engineer. (This requirement is stipulated to avoid circumventing the intent of the *I&RTS* by loading the runtime segment as directed by the installer, and then moving the segment to some other location during PostInstall**{** XE "PostInstall" **}**.) This requirement does *not* apply to COTS segments or to patch segments - as approved by the Chief Engineer - that must move files into the segment being patched, nor does it apply to data that is being moved to the proper UNIX $DATA_DIR**{** XE "DATA_DIR" **}{** XE "$DATA_DIR" **}**/global or $DATA_DIR/local directories.
T	**F**	**N/A**	**5-24**	(NT) For full segmentation ~~segments~~, the segment creates all its subkeys underneath *SegType***{** XE "SegType" **}**\*SegDir* where *SegType* is COE, COTS, Patches, Data, Database, or Software, and *SegDir* is the segment's directory name.
T	**F**	**N/A**	**5-25**	(NT) Except for COTS, ~~segment~~ top-level registry keys are named with the segment prefix.
T	**F**	**N/A**	**5-26**	(NT) The segment installation defaults to the current setting of the registry key HKLM\Software\Microsoft\Windows\CurrentVersion\ProgramFilesDir.
T	**F**	**N/A**	**5-27**	(NT) The segment is fully compliant with the *Designed for Microsoft Windows NT and Windows 98 Logo Handbook for Software Applications* requirements as tailored by the Windows New Technology (NT)-Based Segments chapter.
				Operating System Services
T	**F**	**N/A**	**5-28**	The segment does not rename well-defined ports (e.g., ftp, ping, and ports less than 1024), or declare new port names that have the same port number as well-defined ports.
T	**F**	**N/A**	**5-29**	If ports are required, they have been identified and documented in the COEServices**{** XE "COEServices" **}** segment descriptor.
				GUI Environment
T	**F**	**N/A**	**5-30**	The segment is fully compliant with the style of the native GUI (see compliance requirements in the *DII COE User Interface Specifications***{** XE "*User Interface Specification*" **}**).
T	**F**	**N/A**	**5-31**	The segment, if it requires a window manager in the target system, uses the window manager provided by the COE ~~(dtwm for UNIX, Windows NT window manager for NT platforms)~~.
T	**F**	**N/A**	**5-32**	(UNIX) The segment is compatible with the XFONTSDIR**{** XE "XFONTSDIR" **}**, XAPPLRESDIR**{** XE "XAPPLRESDIR" **}**, and XENVIRONMENT**{** XE "XENVIRONMENT" **}** settings established by the COE. A segment may change the settings of these environment variable**{** XE "environment variable" **}**s as long as they are in effect *only* for the segment's local environment and do not affect the global execution environment.

Database Services				
T	F	N/A	5-33	Application segments are separate from their corresponding database segment.
T	F	N/A	5-34	Application segments should operate correctly on both client workstations and data servers (i.e., application segments should not have to be installed on a database server).
T	F	N/A	5-35	~~The segment i~~Installation revokes the owner account's DBMS login privilege upon successful completion of database installation so that no owner accounts can be used by a user to connect to the database.
T	F	N/A	5-36	Database owner accounts do not have database administrator privileges.
T	F	N/A	5-37	Separate segments are provided that create required database dependencies. These segments are executed by the owning database(s).
T	F	N/A	5-38	Segments do not modify the core DBMS instance's configuration provided by the DII COE or other systems, but may use them as a template to create other DBMS instance~~ segment~~s.
T	F	N/A	5-39	The database owner account must use the segment prefix to ensure its uniqueness within the COE community.
T	F	N/A	5-40	DB segment users have unique accounts in those DB segments.
T	F	N/A	5-41	The segment does not assume any particular disk configuration when creating data files.
T	F	N/A	5-42	~~Segment d~~Dependencies on DBMS vendor-supplied products that are not provided by the COE DBMS segments are documented in the IntgNotes{ XE "IntgNotes" } file.
T	F	N/A	5-43	Any modified versions of DBMS COE tools reside with the application's client segment.
T	F	N/A	5-44	The Database segment does not use raw partitions.
T	F	N/A	5-45	The segment does not modify database schema objects which have been created by another segment (e.g.. columns cannot be added/changed/deleted). If the segment modifies another segment's database by adding roles, triggers, views, database procedures, constraints, indices, etc., those modifications shall be documented with the appropriate Requires and Database descriptors and coordinated with the cognizant Chief Engineer or DOD configuration authority.
T	F	N/A	5-46	The database segment can coexist on a shared DB server without corrupting another segment's data.
T	F	N/A	5-47	Grants are not made to public or general-purpose users (e.g. Oracle's PUBLIC user).
T	F	N/A	5-48	Only the owner of the database objects and the DBA are able to administer grants.
T	F	N/A	5-49	Operations that set or redirect the user's DBMS environment variable{ XE "environment variable" }s take place only within the application's execution space.

T	**F**	**N/A**	**5-50**	Application-level permissions are not granted to DBA accounts or to database roles used for DBMS administration.
T	**F**	**N/A**	**5-51**	The database segment does not alter the database roles that are part of the COTS DBMS.
T	**F**	**N/A**	**5-52**	Universal and shared database segments provide generic "read-only" roles.
T	**F**	**N/A**	**5-53**	Database roles are not granted to DBAs.
T	**F**	**N/A**	**5-54**	Database segments are identified as Universal, Unique, or Sharable according to their potential for sharing through the `$SCOPE`**{** XE "`$SCOPE`" **}** keyword.
T	**F**	**N/A**	**5-55**	Proposed changes to the DISA-defined configuration of the COTS DBMS segments have been approved by the DII COE Chief Engineer.
T	**F**	**N/A**	**5-56**	Database segments that use specialized DBMS services which affect other database segments have been approved by the cognizant Chief Engineer.
T	**F**	**N/A**	**5-57**	The segment utilizes segment-owned database storage areas. It does not modify or create objects in the core database storage areas (e.g., Oracle System tablespace). Descriptive comments about the segment's data objects are an allowable exception.
T	**F**	**N/A**	**5-58**	The `ReleaseNotes` file of the database segment contains a description of its database objects including, at a minimum, tables, elements, indexes, privileges, and triggers, or references an external document that describes the database objects.
T	**F**	**N/A**	**5-59**	The `ReleaseNotes` file of the database segment contains a description of the storage structure of the segment, or references an external document that describes the storage structure.
T	**F**	**N/A**	**5-60**	No predefined datastores are included in a database runtime segment (e.g., the `DBS_files` directory is empty on the `MakeInstall` version of the database runtime segment).
T	**F**	**N/A**	**5-61**	All deliberately disabled constraints are documented in the segment's `ReleaseNotes` descriptor file, either explicitly, or by reference to another document.
Web Services				
T	**F**	**N/A**	**5-62**	The Web segment supports HTML 3.2 and complies with style specifications (see the *DII COE User Interface Specifications***{** XE "*User Interface Specification*" **}**) for Web applications.
T	**F**	**N/A**	**5-63**	The Web segment provides a notification to "disadvantaged" users if they are using a browser that does not support the features provided by the segment.
T	**F**	**N/A**	**5-64**	(UNIX) The Web segment's application HTML files that are used at runtime are in the segment's `$DATA_DIR`**{** XE "`DATA_DIR`" **}{** XE "`$DATA_DIR`" **}**`/local/`*`SegDir`*`/pub` directory.

Runtime Environment				
T	**F**	**N/A**	**5-65**	~~T~~If the runtime segment is operator-initiated, it is launched ~~from~~ using ~~the~~a ~~same~~ desktop provided with the COE.
T	**F**	**N/A**	**5-66**	The desktop is configured in accordance with the *DII COE User Interface Specifications*{ XE "*User Interface Specification*" }.
T	**F**	**N/A**	**5-67**	The runtime segment uses pathnames relative[4] to the segment's home directory for files within the segment that support the installation/deinstallation process (e.g., `PreInstall`, `PostInstall`, or `DEINSTALL`) ~~-~~so that the installer may choose where to load the runtime segment.
T	**F**	**N/A**	**5-68**	The runtime segment does not alter any reserved symbols from the *I&RTS* Runtime Environment chapter and the Windows New Technology (NT)-Based Segments chapter, unless approved to do so by the DII COE Chief Engineer.
T	**F**	**N/A**	**5-69**	The runtime segment does not override or alter the value of any environment variable[5]{ XE "environment variable" } that it doesn't create.
T	**F**	**N/A**	**5-70**	The segment completely separates the development environment from the runtime environment, and no development environment tools, scripts, or other executables are required at runtime.
T	**F**	**N/A**	**5-71**	The runtime segment uses the same global runtime environment configuration as provided by the COE, or for realtime systems, as configured by a DII COE-approved ~~;~~process; extensions, if any, are made through the appropriate segment descriptors.
T	**F**	**N/A**	**5-72**	The runtime segment only listens on assigned ports, only registers assigned RPC addresses, and for UNIX, only adds assigned system UIDs.
T	**F**	**N/A**	**5-73**	The runtime segment is not tied to a particular server or workstation name (i.e., the runtime segment does not hardcode a server or workstation name).
T	**F**	**N/A**	**5-74**	(UNIX) The runtime segment does not use the "~" character for referencing pathnames which become a part of the global runtime environment.

[4] The purpose of this requirement is to simultaneously support the need to choose where segments are loaded at installation time rather than being hardcoded, and to avoid potential security vulnerabilities caused by relative pathnames. In concept, the solution is to dynamically determine where the segment is located after it is installed and then use that absolute path to locate desired files or directories. For example: (1) set PATH_ROOT = findmyseg() (2) set fileopen = open(PATH_ROOT/myfile).

[5] This requirement also includes modifying the *global* setting of the `CLASSPATH` environment variable for Java applications. As explained in Chapter 8, the cognizant Chief Engineer may grant interim approval to modify the `CLASSPATH` environment variable as an aid in migration of legacy Java code. If practical, the cognizant Chief Engineer should require that the `CLASSPATH` modification be a *local* setting only so as not to adversely impact other segments in the system.

T F N/A 5-75	(NT) The runtime segment uses relative pathnames, shortcuts, or registry entries to reference files within the runtime ~~segment that support~~segment that supports the installation/deinstallation process.
T F N/A 5-76	(NT) The segment's registry entry points to the segment's current location on disk.
T F N/A 5-77	(NT) The runtime segment does not override or alter the value of any registry key that it doesn't create, except as approved in the Windows NT Logo Handbook.
T F N/A 5-78	(NT) The runtime segment does not create any groups with the same functions as pre-existing groups.
T F N/A 5-79	(NT) In accordance with Microsoft Logo requirements, runtime segment installation properly updates the registry key `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall`.
COE-Component Segments	
T F N/A 5-80	The segment has been approved as a COE-component segment by the DII COE Chief Engineer.
T F N/A 5-81	Segments in the COE kernel fully specify dependencies upon supporting components within the kernel. This is done through the `Requires`**{** XE "`Requires`" **}** segment descriptor.
T F N/A 5-82	All executables and other public symbols use the segment prefix unless otherwise approved by the DII COE Chief Engineer. (Certain legacy segments may be "grandfathered" by the DII COE Chief Engineer. COTS products cannot generally conform to this requirement. Public symbols within COTS products for which there is no choice are exempted.)
T F N/A 5-83	(UNIX) If a COE child**{** XE "child" **}** runtime segment, the runtime segment does not alter the UNIX `path` environment variable**{** XE "environment variable" **}**. This rule does not apply to the parent COE-component runtime segment and applies only to the runtime environment, not the software development environment. It also does not apply to `PostInstall`, `PreInstall`, or `DEINSTALL` descriptors or other scripts that are used only during the installation/deinstallation process.
Aggregate Segments	
T F N/A 5-84	If a parent segment, the segment does not specify a dependency on any of its child segments.
T F N/A 5-85	If a child segment, the segment does not specify a dependency on any other children in the aggregate.
T F N/A 5-86	Only one segment in the aggregate is designated as the parent.

Segment Descriptors				
T	**F**	**N/A**	**5-87**	The segment describes all background processes, if any, through the `Processes{ XE "Processes" }` descriptor. (If the product is a COTS product that starts its own background processes instead of using the `Processes` descriptor, the processes started must be documented in the *SVD* document or its equivalent, and a waiver granted by the Chief Engineer.)
T	**F**	**N/A**	**5-88**	All segment dependencies and conflicts are fully declared through the appropriate descriptor. Segments shall ~~(Mission-application segments need not specify dependencies on segments contained in the COE kernel unless they are version sensitive. COE-component segments need not~~ specify any direct dependencies they have on components within the COE kerne~~l unless they are sensitive to version changes in the COE kerne~~l.~~)~~
T	**F**	**N/A**	**5-89**	Memory and disk space requirements are fully and accurately specified in the `Hardware{ XE "Hardware" }` descriptor. For abbreviated segmentation, the disk space requirements must be consistent with the abbreviated segmentation approach that was approved by the Chief Engineer. (For example, if the “pseudo-segment” is loaded before the COTS product, the disk space requirement stated is the sum of the space required for the segment descriptors and for the software loaded through vendor-provided installation procedures.)
T	**F**	**N/A**	**5-90**	If not a permanent segment, the `DEINSTALL{ XE "DEINSTALL" }` script and `Comm.deinstall{ XE "Comm.deinstall" }` descriptors ensure (a) that they correctly make the changes indicated, (b) that they completely reverse modifications made by `Community{ XE "Community" }`, `PostInstall{ XE "PostInstall" }`, and `PreInstall{ XE "PreInstall" }` when the segment was loaded, and (c) that they do not inadvertently destroy settings that may have been made by another segment.[6] (The installation tools will not necessarily reverse all modifications made by the `COEServices{ XE "COEServices" }` descriptor as explained in Chapter 6. The segment developer is not required to reverse these modifications.)
T	**F**	**N/A**	**~~5-90~~5-90a**	Requirements for reserving shared memory are specified in the `Hardware` descriptor with the `$OS_SHARED_MEMORY` keyword.
T	**F**	**N/A**	**~~5-90~~5-90b**	Dependencies on specific hardware devices are documented in the `Hardware` descriptor with the `$HW_DEVICE` keyword.
T	**F**	**N/A**	**5-91**	(UNIX) The `PostInstall{ XE "PostInstall" }`, `PreInstall{ XE "PreInstall" }`, and `DEINSTALL{ XE "DEINSTALL" }` scripts have been checked and verified to *not* do a UNIX `mv` across file partitions.
T	**F**	**N/A**	**5-92**	(NT) Unless a COTS segment or an abbreviated segment, the segment uses the `Processes{ XE "Processes" }` descriptor to create boot time processes. It does not directly set the `Run{ XE "Run" }` or `RunOnce{ XE "RunOnce" }` keys underneath

[6] Developers should generally use `$APPEND{ XE "$APPEND" }` to add to community files{ XE "community files" }, rather than `$DELETE{ XE "$DELETE" }` or `$REPLACE{ XE "$REPLACE" }`. Developers should ensure that they delete or replace only those entries to a community file that their segment would have added.

	CurrentVersion but relies upon the installation tool to do so{ XE "CurrentVersion" }.
T F N/A ~~5-92~~5-92a	(Realtime) The segment fully specifies dependencies upon operating system units of functionality. This is done through the RequiredServices descriptor with the $POSIX_UNIT keyword.

Process Compliance				
T	**F**	**N/A**	**5-93**	The segment has been registered with the appropriate SSA.
T	**F**	**N/A**	**5-94**	The Chief Engineer has granted prior approval for background, boot, RunOnce{ XE "RunOnce" }, privileged, and periodic processes.
T	**F**	**N/A**	**5-95**	The segment prefix and segment name used are the prefix and name assigned at segment registration time.
T	**F**	**N/A**	**5-96**	System resources required by the segment have been registered with the SSA.
T	**F**	**N/A**	**5-97**	The ports, UIDs (UNIX), GIDs (UNIX), named pipes, and RPC logical addresses (i.e., the symbolic name passed to `rpcbind` for transport-independent RPC implementations, or the actual port for socket-based RPC implementations) being used are those assigned at segment registration time.
T	**F**	**N/A**	**5-98**	The platforms and operating systems on which the segment can run have been identified and documented in the *SVD* document, or its equivalent.
T	**F**	**N/A**	**5-99**	All COTS products required, including the required version, are documented in the *SVD* document or its equivalent.
T	**F**	**N/A**	**5-100**	All required licenses are provided to the SSA with the segment, or negotiations have been made with the SSA to use licenses procured by the SSA.
T	**F**	**N/A**	**5-101**	Segment dependencies are noted in the *SVD* document or its equivalent.
T	**F**	**N/A**	**5-102**	The *SVD* document, or its equivalent, has been submitted with the segment to the SSA.
T	**F**	**N/A**	**5-103**	The segment has been submitted to and accepted for inclusion in the SSA’s online library.
T	**F**	**N/A**	**5-104**	The `VERSION`{ XE "`VERSION`" } descriptor has been updated from the previous release in accordance with the requirements specified in the Runtime Environment chapter. (This does not apply to the initial release of the segment.)
T	**F**	**N/A**	**5-105**	Whether full or abbreviated segmentation is used, the segment successfully passes `VerifySeg` with no errors and the segment is submitted with an annotated output from `VerifySeg`{ XE "`VerifySeg`" }. All warnings are explained in full in `VSOutput`{ XE "`VSOutput`" }.[7]
T	**F**	**N/A**	**5-106**	The segment is submitted with a set of integration notes (`IntgNotes`{ XE "`IntgNotes`" }) as described in the Runtime Environment chapter.
T	**F**	**N/A**	**5-107**	The runtime segment has been loaded and tested in a “clean” COE environment prior to submission to the SSA.

[7] This can be done by redirecting the output of `VerifySeg`{ XE "`VerifySeg`" } to the file `VSOutput`{ XE "`VSOutput`" }. Then, use any convenient ASCII editor to edit `VSOutput` to insert comments to explain all warning messages.

T	**F**	**N/A**	**5-108**	Runtime s~~S~~egment installation has been tested through the same installation tools used by site operators. (`TestInstall`**{** XE "`TestInstall`" **}** alone does *not* satisfy this requirement. The `COEInstaller`**{** XE "`COEInstaller`" **}** tool must be used to load and remove the runtime segment.)
T	**F**	**N/A**	**5-109**	If removable, the runtime segment has been tested and confirmed that it can be successfully removed from the system.
T	**F**	**N/A**	**5-110**	If special installation/integration procedures/problems exist, then they are incorporated into the `PostInstall`**{** XE "`PostInstall`" **}** (or other) descriptors as appropriate, and documented in the `IntgNotes`**{** XE "`IntgNotes`" **}** descriptor file.
Miscellaneous				
T	**F**	**N/A**	**5-111**	The runtime segment creates and initializes any dynamic data files it contains that are updated as the system executes (e.g., message logs, and operator preferences). If an expected file is missing, the runtime segment generates a runtime error message and gracefully terminates with an appropriate message to the operator.
T	**F**	**N/A**	**5-112**	If a patch segment, it follows the patch segment naming convention.
T	**F**	**N/A**	**5-113**	The segment does not alter any files outside its own directory with the following exceptions: (a) the segment is a patch segment and must modify files in another segment; (b) the segment is creating temporary files or directories in directories established for temporary storage; (c) the segment is modifying files created for it by the operating system; (d) the files are created or modified through approved APIs or segment descriptors during segment installation; or (e) the files are owned by the segment but reside in approved COE locations outside the segment's assigned directory (e.g., `/h/data/local`, `/h/data/global`, user directories).
T	**F**	**N/A**	**5-114**	The segment does not create copies of executables from other segments. (There are rare instances where this may be required to create a patch segment. Such exceptions require the prior approval of the Chief Engineer.)
T	**F**	**N/A**	**5-115**	The segment does not contain any circular dependencies (e.g., Seg A depends on Seg B, Seg B depends on Seg C, Seg C depends on Seg A is not allowed).
T	**F**	**N/A**	**5-116**	For full segmentation~~ segments~~, the runtime segment does not delete itself via the `DEINSTALL`**{** XE "`DEINSTALL`" **}** descriptor, nor perform any other operations that are handled by the COE installation tools (e.g., undo changes made to community files**{** XE "community files" **}**). This requirement also applies to the "pseudo-segment" created as part of the abbreviated segmentation process.
T	**F**	**N/A**	**5-117**	(NT) If the segment creates a new file extension, the extension has been registered with the cognizant Chief Engineer.

B-6. Intermediate DII COE Compliance (Level 6)

Security Services				
T	**F**	**N/A**	**6-1**	If the data for a particular segment contains any classified entries, then all of its data is packaged in a separate data segment and classified accordingly.
T	**F**	**N/A**	**6-2**	Classified segments are packaged separately from unclassified segments, or from segments which are classified at a lower level. (It is permissible to create aggregate segments that contain segments at different classification levels, but the aggregate must be labeled with the highest classification level of any segment within the aggregate.)
T	**F**	**N/A**	**6-3**	Termination of runtime segment execution, whether premature, inadvertent, or intentional does not place the operator at a command-line prompt.
T	**F**	**N/A**	**6-4**	Privileged processes, if required, have been approved by the Chief Engineer and are listed in the `Processes`**{** XE "`Processes`" **}** segment descriptor. (If the product is a COTS product that starts its own background processes instead of using the `Processes` descriptor, the processes started must be documented in the *SVD* document or its equivalent, and a waiver granted by the Chief Engineer.)
T	**F**	**N/A**	**6-5**	For runtime segments, n~~N~~o directory or file permission, whether created at install time or runtime, is less restrictive than identified in the Security chapter's directory/file permissions table, unless approved by the Chief Engineer. Any ~~such~~ directories or files that do not meet the permissions identified in the table, whether for an extended toolkit or a runtime segment, are documented in the *SVD* document or its equivalent. `VerifySeg`**{** XE "`VerifySeg`" **}** also uses this table to check permissions. All security-related messages from `VerifySeg` are explained in the `VSOutput`**{** XE "`VSOutput`" **}** file and documented in the *SVD* document or its equivalent.
T	**F**	**N/A**	**6-5a**	If the runtime segment uses a public/private key infrastructure, it uses the DOD PKI standard unless ~~a waiver has been approved by the cognizant Program Manager~~the cognizant Program Manager has approved a waiver.
T	**F**	**N/A**	**6-6**	(NT) If the runtime segment creates groups, the groups follow the naming conventions in the Windows New Technology (NT)-Based Segments chapter.
Standards Compliance				
T	**F**	**N/A**	**6-7**	The segment is either completely compliant with the *DII COE User Interface Specifications***{** XE "*User Interface Specification*" **}** or has minimal[8] deviations that have been approved by the Chief Engineer.

[8] "Minimal" is defined in the *Style Guide* and in the absence of *Style Guide* direction, at the discretion of the Chief Engineer.

T F N/A 6-8	The segment is available on all COE-supported platforms unless otherwise approved by the Chief Engineer.

T	F	N/A	6-9	The runtime segment does not alter any community files{ XE "community files" } except through COE segment descriptors or published APIs. (This does not apply to vendor-supplied installation software used in an abbreviated segmentation process.)
T	F	N/A	6-10	The segment does not use directories with different names than specified in the Runtime Environment chapter and the Windows New Technology (NT)-Based Segments chapter to fulfill the purpose of Scripts, bin, data, etc.
T	F	N/A	6-11	If the ~~segment~~ extended toolkit contains APIs written in C, the header files for the public APIs are ANSI-C-compliant, ~~and~~ use function prototypes, and the header files are constructed to support C++ calling routines.
T	F	N/A	6-12	All XML documents and their related XML elements that are used in a Public Interface are fully documented and registered in the DII COE XML Registry.
T	F	N/A	6-13	(NT) Except for COTS, the segment stores private INI files, if any, in the segment's data\INI subdirectory.
T	F	N/A	6-14	(NT) The segment uses filename extensions in accordance with standard Windows usage (TXT for ASCII files, DLL for dynamic link libraries, etc.).
GUI Environment				
T	F	N/A	6-15	(UNIX) The segment does not alter any X or Motif supplied files (e.g., Xdefaults, rgb.txt).
Database Services				
T	F	N/A	6-16	If PUBLIC synonyms are used, they start with the segment prefix to guarantee uniqueness.
T	F	N/A	6-17	Database runtime segments do not create user accounts, except for database owner accounts. Scripts are provided for the DBA's use to add, modify, and remove user privileges, roles, access control, etc. These scripts are documented in the IntgNotes{ XE "IntgNotes" } file.
T	F	N/A	6-18	Grants are made to database roles/groups, not user accounts or general-purpose users (e.g., Oracle's PUBLIC user).
T	F	N/A	6-19	The ~~application~~ runtime segment does not assume the existence of any particular database user account.
T	F	N/A	6-20	The database segment documents all dependencies on other database segments in the Requires{ XE "Requires" } and Database{ XE "Database" } sections of the SegInfo{ XE "SegInfo" } descriptor file.
T	F	N/A	6-21	The segment uses only the DBMS provided by the COE, or has a migration plan approved by the cognizant Chief Engineer.
T	F	N/A	6-22	The DBMS is not tuned for a specific application.
T	F	N/A	6-23	Applications connect to the DBMS through published APIs.
T	F	N/A	6-24	Data objects and elements follow naming conventions specified in the DII COE Data Concepts chapter.

T	**F**	**N/A**	**6-25**	Descriptive definitions for tables, elements, and views are stored in the system's data dictionary tables as comments less than 255 characters.
T	**F**	**N/A**	**6-26**	Database segments are fully documented with descriptions of their database schema, business rules, valid values, element definitions and other features, preferably in an IDEF1X or UML data model. The ReleaseNotes file of the database segment should contain the descriptions or reference external document(s) that provide them.
T	**F**	**N/A**	**6-27**	Vertical views do not contain hidden columns.
T	**F**	**N/A**	**6-28**	CREATE table statements explicitly stipulate NOT NULL or NULL constraints for every column for all tables.
T	**F**	**N/A**	**6-29**	If there are multiple owners associated with a database, a separate script to create the database object is provided for each owner.
T	**F**	**N/A**	**6-30**	The database segment does not create objects that are owned by a DBMS default account.
T	**F**	**N/A**	**6-31**	If the runtime segment contains a large static database, it is provided as a separate data segment.
T	**F**	**N/A**	**6-32**	(UNIX) The database runtime segment creates its data stores through the PostInstall descriptor file, preferably through the database administration runtime tools.
T	**F**	**N/A**	**6-33**	(NT) Database files used at runtime are on the data drive as indicated in the DII COE logical drive layout, or the Chief Engineer has approved a migration plan to achieve the DII COE logical drive layout.
Web Services				
T	**F**	**N/A**	**6-34**	The runtime segment uses a Web server provided by the COE rather than bringing along its own Web server.
T	**F**	**N/A**	**6-35**	(NT) Runtime s~~S~~egment web files (e.g., HTML, Active Server Pages) are on a dedicated logical drive in accordance with the DII COE logical drive layout, or the Chief Engineer has approved a migration plan to achieve the DII COE logical drive layout.
COTS Products				
T	**F**	**N/A**	**6-36**	The FilesList{ XE "FilesList" } descriptor has been validated as correctly documenting what files and directories constitute the COTS product. (The FilesList descriptor need not include the SegDescrip{ XE "SegDescrip" } directory or its files.) This does not apply to COTS products in the COE kernel, but does apply to all abbreviated segmentation files whether the products are COTS or not.
T	**F**	**N/A**	**6-37**	(UNIX) The PostInstall{ XE "PostInstall" } or PreInstall{ XE "PreInstall" } script ensures that there is enough space in the directories where the COTS product will be installed and uses COEInstError{ XE "COEInstError" } to report an error message if not. This requirement applies to the DEINSTALL script also if it is possible to run out of disk space when DEINSTALL is executed as part of upgrading an installed segment.

T	**F**	**N/A**	**6-38**	(UNIX) All COTS products are packaged as separate, individual COTS segments, unless approved by the cognizant Chief Engineer.
T	**F**	**N/A**	**6-39**	(NT) All COTS products are packaged as separate, individual COTS segments unless approved by the cognizant Chief Engineer.
				Runtime Environment
T	**F**	**N/A**	**6-40**	If the runtime segment creates temporary files, they are deleted when no longer needed.
T	**F**	**N/A**	**6-41**	If the runtime segment uses absolute pathnames to reference files outside the segment, it is able to determine the absolute path at runtime. For UNIX segments (including extended toolkits and runtime segments), the segment is able to handle symbolic links that are themselves symbolic links.
T	**F**	**N/A**	**6-42**	The runtime segment does not create any *global* environment variable**{** XE "environment variable" **}**s[9] or other public symbols with the same name as any environment variables listed as reserved in the *I&RTS*.
T	**F**	**N/A**	**6-43**	Shared libraries provided by UNIX segments are in the segment's `bin` subdirectory. For UNIX shared libraries and NT DLLs, the `SharedFile`**{** XE "`SharedFile`" **}** descriptor is used to define them, and they are named using the segment prefix convention.
T	**F**	**N/A**	**6-44**	If the segment provides public APIs, all uses of signals and process or thread creation within the segment's public libraries are documented in the appropriate programmer's guides. Moreover, all such API functions shall be reentrant to allow them to be called from a multithreaded application.
T	**F**	**N/A**	**6-45**	(UNIX) The runtime segment reuses environment variable**{** XE "environment variable" **}**s already defined by the COE. It does not create any environment variables that are identical in value to those defined by the COE, or that can be derived from them.
T	**F**	**N/A**	**6-46**	(UNIX) For runtime segments, fonts and app-defaults located underneath the segment's `data` subdirectory follow the segment prefix naming convention specified in the *I&RTS*.
T	**F**	**N/A**	**6-47**	(UNIX) The runtime segment appends, not prepends, its `bin` subdirectory to the environment variable**{** XE "environment variable" **}** used for the search path for finding executables. (This does not apply to COE child segments.)
T	**F**	**N/A**	**6-48**	(NT) For full segmentation, the segment stores its DLL files in the segment's `bin` subdirectory.
T	**F**	**N/A**	**6-49**	(NT) Unless a COTS segment, the segment does not alter the globally defined Windows system `path` environment variable**{** XE "environment variable" **}**.
T	**F**	**N/A**	**6-50**	(NT) The runtime segment uses the specified DII COE logical drive layout, or the Chief Engineer has approved a migration plan

[9] This requirement applies only to affecting the *global* environment that another segment may inherit. Segments may set their local environment, which is not inherited by other segments and hence does not impact them, to whatever is appropriate for the segment's design.

	to achieve the DII COE logical drive layout.

T	**F**	**N/A**	**6-51**	(NT) All executable and DLL file version properties are completed in accordance with the Windows New Technology (NT)-Based Segments chapter.
Segment Descriptors				
T	**F**	**N/A**	**6-52**	The `ReleaseNotes`{ XE "`ReleaseNotes`" } descriptor conforms to the requirements stipulated in the Runtime Environment chapter and the Windows New Technology (NT)-Based Segments chapter.
T	**F**	**N/A**	**6-53**	If any runtime segment files need special permission/ownership settings, they are established through the `FileAttribs`{ XE "`FileAttribs`" } descriptor if the descriptor supports the required setting. Exceptions to this are documented and approved by the Chief Engineer.
Process Compliance				
T	**F**	**N/A**	**6-54**	The segment includes a comprehensive API test suite that exercises all APIs provided by the segment.
T	**F**	**N/A**	**6-55**	The extended toolkit ~~segment~~ includes man pages, help files, or HTML-format pages, for all APIs that are to be distributed with the ~~Developer's Toolkit associated with the segmen~~extended ~~t~~toolkit.
T	**F**	**N/A**	**6-56**	The extended toolkit ~~segment~~ has been compiled without the debug option enabled and delivered to the SSA. A developer *may* deliver an extended toolkit in both a debug enabled and a debug disabled version, but *shall* deliver a version without debug enabled. The runtime segment *shall* use the version with debug disabled.~~.~~
T	**F**	**N/A**	**6-57**	~~If the segment has published APIs implemented as shared libraries, static libraries are provided as well.~~ **This requirement has been deleted.**
T	**F**	**N/A**	**6-58**	**This requirement has been deleted.**~~If the segment uses another segment's public APIs and they are implemented as shared libraries, the segment is submitted linked with the shared libraries and not the static libraries.~~
T	**F**	**N/A**	**6-59**	If the segment has a `DEINSTALL`{ XE "`DEINSTALL`" } and `Community`{ XE "`Community`" } descriptor, it also includes a `Comm.deinstall`{ XE "`Comm.deinstall`" } descriptor which reverses the actions of the `Community` descriptor during segment removal.
T	**F**	**N/A**	**6-60**	The segment has been tested to ensure that it successfully installs over and replaces any previous version of the segment.
T	**F**	**N/A**	**6-61**	(UNIX) The segment executables have been run through the UNIX `strip` program prior to installation in the target environment. This requirement does not apply to COTS product~~segment~~s. For runtime segments, this means that the `strip` program must be run prior to `MakeInstall`. For extended toolkits, this means that the system integrator must run `strip` prior to installation in the target runtime environment.

Miscellaneous	
T F N/A 6-62	If the COE provides functions required by the segment, at least 50% of the functions required are provided by the COE and not by duplicative code in the segment.
T F N/A 6-63	API backward compatibility conforms to the version numbering principles described in the version numbering section of the DII COE chapter.
Java Conventions	
T F N/A 6-64	The segment conforms to the Java naming conventions specified in Chapter 8 for Java classes, packages, method names, and interface names.
T F N/A 6-65	The segment does not use company name to name packages, interfaces, classes, or methods. This does not apply to COTS products.
T F N/A 6-66	The segment's jar files that are stored outside the segment use the segment prefix to ensure that the segment's jar filenames do not conflict with other segments.
T F N/A 6-67	The segment's jar files are named with the `.jar` extension (or `.zip` for JDK-1.1 or earlier).
T F N/A 6-68	The segment does not store any classified class or jar files in the DII COE shared jar/class directory.
T F N/A 6-69	The segment does not replace COE-provided standard Java runtime tools, class libraries, or jar files.
T F N/A 6-70	If the segment requires a modified version of standard Java class libraries or jar files, the segment has completely encapsulated the modifications in its own segment.
T F N/A 6-71	The segment declares methods that are not accessible outside the class to be `private`. Segment classes that are not intended to be extended are declared `final`.
T F N/A 6-72	The runtime segment does not change system properties that are global to the current JVM instance.

B-7. Interoperable Compliance (Level 7)

Security Services				
T	**F**	**N/A**	**7-1**	The segment does not place any temporary files in the system maintained temporary directory that are sensitive to alteration, deletion, or disclosure to unauthorized users.
T	**F**	**N/A**	**7-2**	If the segment creates files that are sensitive to alteration or deletion by unauthorized users, they are not placed in any directory where such unauthorized users have write access, and those files do not have write permissions set for such unauthorized users.
T	**F**	**N/A**	**7-3**	If the segment creates files that are sensitive to disclosure to unauthorized users, they are not placed in any directory where unauthorized users have read access.
T	**F**	**N/A**	**7-4**	The segment does not contain features with multiple security levels, unless an aggregate segment.
T	**F**	**N/A**	**7-5**	Unclassified sample data is provided with the segment to allow for unclassified testing and training.
T	**F**	**N/A**	**7-6**	(UNIX) Data files with different file permissions are split into separate directories underneath the segment's `data` subdirectory.
T	**F**	**N/A**	**7-7**	(NT) The segment will not install unless being installed by a user with administrative privileges.
T	**F**	**N/A**	**7-8**	(NT) If the segment utilizes the `Run` or `Find` dialog, the segment provides an alternate method to accomplish the required task when a system policy is in effect that disables the user's capability to use these features.
Standards Compliance				
T	**F**	**N/A**	**7-9**	If written in C, the segment is ANSI-C-compliant.
T	**F**	**N/A**	**7-10**	If the segment contains public APIs, C, C++, or IDL interfaces are provided unless the Chief Engineer grants a waiver for a different style interface (e.g., by defining a file format).
T	**F**	**N/A**	**7-11**	Global and local data owned by the segment are located underneath `$DATA_DIR`**{** XE "`DATA_DIR`" **}{** XE "`$DATA_DIR`" **}** as described in the Runtime Environment chapter for UNIX and as described in the Windows New Technology (NT)-Based Segments chapter for NT.
T	**F**	**N/A**	**7-12**	For runtime segments, o~~O~~perator-specific data is located underneath `/h/USERS` as described in the Runtime Environment chapter for UNIX and as described in the Windows New Technology (NT)-Based Segments chapter for NT.
T	**F**	**N/A**	**7-13**	Excepting COTS ~~segments~~products, all environment variable**{** XE "environment variable" **}**s and top-level registry keys are named with the segment prefix unless approved by the Chief Engineer. (The Chief Engineer may approve "grandfathering" certain environment variables.)

T F N/A 7-14	All XML documents and their related XML elements that are used in a Public Interface have been approved by the relevant Community of Interest listed in the DII COE XLM Registry.
T F N/A 7-15	(UNIX) The segment uses only POSIX.1-defined interfaces to access the operating system, unless approved by the Chief Engineer.
GUI Environment	
T F N/A 7-16	The segment supports cut and paste between GUI-based segments through the use of a shared clipboard.
T F N/A 7-17	(UNIX) The segment uses resource files to control window behavior rather than hardcoded window behavior attributes.
T F N/A 7-18	(NT) The segment uses TrueType fonts. This requirement does not apply to COTS ~~segments~~products.
Database Services	
T F N/A 7-19	Data objects within the runtime segment do not duplicate those already contained in available database runtime segments, except for performance reasons as approved by the cognizant Chief Engineer.
T F N/A 7-20	Data objects that are common to a community or the enterprise are contained in a separate runtime segment from mission-specific data objects.
T F N/A 7-21	Database roles/groups are specific to application privileges, not general purpose.
T F N/A 7-22	Grants are segregated by owner.
T F N/A 7-23	Data object creation script files follow the specified structure and naming convention as defined in the DII COE Data Concepts chapter.
T F N/A 7-24	The database segment's data store identifiers incorporate the segment prefix and the function of the data store.
T F N/A 7-25	The data objects contained within a database segment are approved by the individual responsible for managing the affected namespace(s).
T F N/A 7-26	The application segment accessing a database segment does not have embedded DBMS vendor runtime libraries or environment variables.
T F N/A 7-27	The database segment provides a reload capability and a non-destructive update capability. If data must be migrated from previous versions of a database segment, then tools and/or procedures (automated or manual) are provided.
T F N/A 7-28	The `Database` runtime segment uses the `DBAdmSCreateDS` tool to create physical data storage. If the `DBAdmSCreateDS` is not available for the COE version being used, the developer uses DBMS vendor tools called from the database definition script under the `CREATE_DATA_STORE` argument.

T	**F**	**N/A**	**7-29**	Legacy views of a database table that are included in the table’s definition script are approved by the cognizant Chief Engineer.
T	**F**	**N/A**	**7-30**	Names of rules created on other schemas identify the inter-database linkage as the rule’s function.
Runtime Environment				
T	**F**	**N/A**	**7-31**	The runtime segment does not include any environment variable**{** XE "environment variable" **}**s that could be derived from an already defined environment variable.
T	**F**	**N/A**	**7-32**	(UNIX) Runtime s~~S~~egment references to global and local data are done through the `$DATA_DIR`**{** XE "`DATA_DIR`" **}{** XE "`$DATA_DIR`" **}** environment variable**{** XE "environment variable" **}**.
T	**F**	**N/A**	**7-33**	(NT) The runtime segment is in full compliance with the specified DII COE logical drive layout.
Miscellaneous				
T	**F**	**N/A**	**7-34**	The segment does not duplicate any functions provided by COE-component segments unless approved by the DII COE Chief Engineer.
T	**F**	**N/A**	**7-35**	No more than 25% of the segment’s accesses to COE-component segments are through private or legacy APIs.

B-8. Full DII COE Compliance (Level 8)

Security Services				
T	**F**	**N/A**	**8-1**	Entry to and exit from the command-line mode causes an entry into the system audit logs that specifies the date, time, and user involved.
T	**F**	**N/A**	**8-2**	Information written to the audit log includes the segment prefix.
T	**F**	**N/A**	**8-3**	The segment does not mix restricted and unrestricted data files in the same directory.
Standards Compliance				
T	**F**	**N/A**	**8-4**	The segment does not use any conventions obsoleted by this document (use of progs vs. bin, use of COMPONENT vs. CHILD, use of ModName**{** XE "ModName" **}** and SegType**{** XE "SegType" **}** vs. SegName**{** XE "SegName" **}** etc.).
T	**F**	**N/A**	**8-5**	All public symbols are named with the segment prefix naming convention.
T	**F**	**N/A**	**8-6**	All directory and file names begin with an alphanumeric character.
T	**F**	**N/A**	**8-7**	(UNIX) The segment follows the convention that data owned by the segment under $DATA_DIR**{** XE "DATA_DIR" **}{** XE "$DATA_DIR" **}** is in the form $DATA_DIR/local/*segdir*/data and $DATA_DIR/global/*segdir*/data where *segdir* is the segment's home directory name.
GUI Environment				
T	**F**	**N/A**	**8-8**	The runtime segment is fully compliant with the *DII COE User Interface Specifications***{** XE "*User Interface Specification*" **}**.
T	**F**	**N/A**	**8-9**	(NT) The runtime segment uses only the Windows print dialog box for selecting printer configuration parameters.
Database Services				
T	**F**	**N/A**	**8-10**	Data elements are chosen from Joint standards (DDM and DDS) and use the data type, field width, and units of measure prescribed in the standard.
T	**F**	**N/A**	**8-11**	A test database is provided together with test procedures to verify correct installation of the database and associated roles, and to verify correct operation of constraints defined in the database. The test data is located underneath the SegDir/Integ/TestSuite.

T	**F**	**N/A**	**8-12**	The runtime segment does not duplicate any data already maintained in the SHADE repository *or* the DII COE-based target system, unless for performance reasons and only as approved by the DII COE Chief Engineer.
T	**F**	**N/A**	**8-13**	The segment uses a DBMS version supported by the DII COE.
T	**F**	**N/A**	**8-14**	All logical data models that identify new data requirements and associated meta-data, domain values, and authoritative sources have been approved against the Defense Data model.
T	**F**	**N/A**	**8-15**	All data mediation required by a segment is provided by that database segment and is consistent with the data requirements documented in the Defense Data Model.
Runtime Environment				
T	**F**	**N/A**	**8-16**	Except for COTS products, all executables and public symbols are named `segprefix_name`, where *segprefix* is the assigned segment prefix.
T	**F**	**N/A**	**8-17**	(UNIX) The runtime segment adds no more than one “home” environment variable**{** XE "environment variable" **}** to the global environment.
Process Compliance				
T	**F**	**N/A**	**8-18**	The segment includes a set of test data for verifying correct segment operation.
Miscellaneous				
T	**F**	**N/A**	**8-19**	The segment does not use any private APIs to access external segments. All accesses are through public APIs or approved protocol standards.
T	**F**	**N/A**	**8-20**	Operator data is located through the appropriate APIs from the ~~Developer’s Toolkit~~extended toolkit documentation for UNIX, and through the appropriate Windows NT API for Windows NT.
T	**F**	**N/A**	**8-21**	The current operator profile is obtained through the appropriate APIs from the ~~Developer’s T~~extended toolkit documentation for UNIX and through the appropriate Windows NT API for Windows NT.
T	**F**	**N/A**	**8-22**	The segment does not duplicate functionality provided by any other segment unless approved by the DII COE Chief Engineer.

B-9. Recommended Guidelines

The items contained in the following checklist are not mandatory and are not considered in establishing the DII COE compliance level.

T F N/A M-1	The runtime segment does not use boot or background processes. It uses session-or-transient-level processes instead. This guideline may not be applicable to all realtime systems.
T F N/A M-2	The segment allows comments in ASCII data files. The # character is the standard for single line comments while C style comments (delimited by the /* */ pair) are the standard for all other comments.
T F N/A M-3	If written in C, the segment has been run through a utility such as `lint` to detect potential coding errors.
T F N/A M-4	If written in C, the segment has been compiled with the `STRICT` constant.
T F N/A M-5	(UNIX) The segment does not use symbolic links.
T F N/A M-6	(UNIX) The segment uses `putenv` or an equivalent technique to create segment-specific environment variable**{** XE "environment variable" **}**s that are inherited locally, rather than adding environment variables to the global environment.
T F N/A M-7	(UNIX) The segment links with X and Motif shared libraries. This guideline may not be applicable to all realtime systems.
T F N/A M-8	(NT) The segment links to DLL functions by using symbolic names, not ordinal numbers.
T F N/A M-9	(NT) The segment exports DLL functions by symbolic name, not ordinal numbers.
T F N/A M-10	(NT) The segment uses the Win32 API GDI for creating 2D graphics.
T F N/A M-11	(NT) The segment uses OpenGL and DirectX APIs for 3D graphics.
T F N/A M-12	(NT) A tool such as `PORTTOOL.EXE` has been used to identify potential problems with how the segment uses Windows APIs.
T F N/A M-13	(NT) The segment operates correctly under both Windows NT and Windows 95/98.
T F N/A M-14	(NT) The recommendations in the *Designed for Microsoft Windows NT and Windows 98 Handbook for Software Development* have been followed.
T F N/A M-15	(NT) The `finalrptx.doc` and `reportx.doc` files created by the Microsoft Logo program `ANALYZER.EXE` are submitted with the segment to the SSA. All exceptions detected by the tool are explained in full in the `VSOutput` file.